ＵＮＩＸ／ＬＩＮＵＸの基本操作

2022 年 10 月 5 日版

西井 淳

目 次

第 1 章　コンソールと X Window System　2
1.1 ウィンドウマネージャ . . . 2
1.2 コンソールの仮想画面への移動 . . . 2
1.3 ウィンドウマネージャ上の仮想画面 . . . 2
1.4 コピー & ペースト . . . 2

第 2 章　UNIX コマンドの基礎　3
2.1 使用上の若干の注意 . . . 3
2.2 ファイル名の補間 . . . 3
2.3 コマンドの起動・中断 . . . 3
2.4 コマンド履歴について . . . 4
2.5 UNIX の基本コマンド (のごく一部) . . . 4
2.5.1 基本操作 . . . 4
2.5.2 ディレクトリ関連 . . . 5
2.5.3 ファイル閲覧・ファイルの情報取得 . . . 5
2.5.4 プロセス管理 (ps, top, kill, killall, xkill) . . . 5
2.5.5 ファイル圧縮 (gzip, bzip2) . . . 6
2.5.6 ファイルを探す (which,locate) . . . 7
2.5.7 文字コードの変換・判定 (nkf) . . . 7
2.5.8 ネットワーク関連のコマンド . . . 8
2.5.9 いろいろな情報を得る . . . 9
2.6 ワイルドカードについて . . . 9
2.7 パイプについて . . . 9
2.8 リダイレクトについて . . . 9
2.9 シンボリックリンク . . . 10
2.10 システムの停止 . . . 10

第 3 章　シェル (Bash)　11
3.1 設定ファイル . . . 11
3.2 エイリアス . . . 11
3.3 パス . . . 12
3.4 シェルスクリプト . . . 12
3.4.1 シェルスクリプトの基本 . . . 12
3.4.2 シェルスクリプトの引数 . . . 13
3.4.3 文字変数 . . . 13
3.4.4 変数の操作 . . . 14
3.4.5 シェルスクリプトの終了 . . . 14
3.4.6 if 文 . . . 14
3.4.7 for 文 . . . 15

3.4.8 case 文 . . . 16
3.4.9 関数 . . . 16

第 4 章 エディタ (emacs) 17
4.1 UNIX 上で動くエディタ . . . 17
4.2 注意事項 . . . 17
4.3 基本操作 . . . 17
4.4 移動 . . . 18
4.5 画面操作 . . . 18
4.6 ファイル一覧窓での操作 . . . 18
4.7 カット/コピー/ペースト . . . 19
4.8 C/C++のプログラムを書く . . . 19
4.9 Emacs のカスタマイズ . . . 19
4.10 LaTeX 文書の作成 (auctex モード)* . . . 20
4.10.1 auctex の使いかた . . . 20
4.10.2 TeX 文書の処理 . . . 20
4.10.3 TeX 文書の作成 . . . 20
4.10.4 TeX 文書の表題表示 . . . 20

第 5 章 ソフトウェアのインストール方法 (apt/rpm) 21
5.1 パッケージ管理について . . . 21
5.2 apt を利用したパッケージのインストール・一斉更新 . . . 21
5.2.1 パッケージの削除 . . . 22
5.2.2 パッケージの情報取得 . . . 22
5.2.3 パッケージとファイルの関係についての情報取得 . . . 22
5.3 rpm を利用したパッケージ管理 . . . 23

第 6 章 システム管理 ** 25
6.1 システム環境の設定 . . . 25
6.2 NFS(Network File System) . . . 25
6.2.1 NFS サーバの設定 . . . 25
6.2.2 NFS クライアントの設定 . . . 26
6.3 autofs を利用したマウント設定 . . . 26
6.4 ちょっとしたセキュリティ向上 . . . 27
6.4.1 外部からアクセスしたいとき . . . 27
6.4.2 余計なサービス (デーモン) 停止 . . . 27
6.5 各ユーザにディレクトリ利用制限 (quota) をつくる . . . 27
6.5.1 quota をかけるパーティションの設定 . . . 28
6.5.2 ユーザに quota をかける . . . 28
6.5.3 quota 状況をみる . . . 28

第 7 章 トラブル!!!* 29
7.1 ログインできない!!!!! . . . 29
7.2 ネットワークに接続できない？ . . . 29
7.3 固まった!!!!! . . . 30

索引 30

はじめに

この原稿では、UNIX/Linux の基本的な操作方法をごく簡単に説明しています。ただし，Vine Linux 6.4 に準拠して書いてますので、それ以外の UNIX/Linux 環境では若干異なる場合もあります。例えば，UNIX コマンドは Mac OS X でもほぼ共通に使えますが、コマンドオプションが若干違う場合もあります。

また、文章はいろいろな原稿のつぎはぎで作っているので，文体が不揃いですが御容赦を。

キー表示について

本稿では、キーの表示は以下のように行っている。

> `C-{xf}`は、Ctrl キーを押しながら x と f を順に押す

ことを示し、

> `C-x k` は、Ctrl キーを押しながら x をまず押し、次に Ctrl キーを離して k を押す

ことを示します。また、

> `M-c` は、**Meta キー** (PC 用キーボードでは通常 Alt, Mac 用キーボードでは Command キー) を押しながら c を押す

ことを示す。

このドキュメントの著作権について

1. 本稿の著作権は西井淳 `nishii@sci.yamaguchi-u.ac.jp` が有します。
2. 非商用目的での複製は許可しますが、修正を加えた場合は必ず修正点および加筆者の氏名・連絡先、修正した日付を明記してください。また本著作権表示の削除は行ってはいけません。
3. 本稿に含まれている間違い等によりなんらかの被害を被ったとしても著者は一切責任を負いません。

間違い等の連絡や加筆修正要望等の連絡は大歓迎です。

第1章　コンソールとX Window System

UNIX では、**コンソール画面**と一般に言われる、基本的に文字しか表示できない画面と、X Window System というグラフィカルな画面がある。Linux を起動すると通常自動で X Window System が起動してログインウィンドウが表示されるが、後述するようにコンソール画面での作業も可能である。

1.1　ウィンドウマネージャ

X Window 上では、複数のウィンドウの管理をしたり、さまざまなデスクトップ環境を提供するシステムとして、**ウィンドウマネージャ**と呼ばれるシステムが起動される。Vine Linux 6.x の標準ウィンドウマネージャは **GNOME** (上部にメニュー) であるが、他にもいろいろあり、高機能な **KDE** も有名である。KDE は

```
# apt-get install task-kde
```

でインストールでき、**ログイン時のメニューでどれを利用するか選択**できる。(`apt-get` については 5.2 参照)

1.2　コンソールの仮想画面への移動

作業はほとんどの場合 X Window 上で行なったほうが便利であるが、X Window 上で障害がおきたときにはコンソール画面にログインして対応をする。

X Window を利用しているときにコンソールに移るキーコマンドは`{CM}-F1` (コントロールキーとメタキーを同時に押しながら F1 を押す) である。X Window に戻るには`{CM}-F7` とする。

Linux のコンソールでは、6 画面の**仮想画面**を使える。コンソール上で `M-F1`, `M-F2`,...,`M-F6` で、画面の切替えをできる。ある画面がキーを受け付けなくなったときには、別の仮想画面に移って救出作業をすればよい。X-Window 上から各コンソール画面へうつるためのキーコマンドは`{CM}-F2`,...`{CM}-F6` である。

1.3　ウィンドウマネージャ上の仮想画面

ウィンドウマネージャの多くも**仮想画面**をサポートしている。KDE の場合は `C-F1`, `C-F2`,.. で他の仮想画面にうつれる。

1.4　コピー & ペースト

X Window 上のウィンドウで表示されている文字は、ほとんどの場合マウスを用いて**コピー** & **ペースト**を行える。マウスの左ボタンを押しながらコピーしたい領域を選択し、コピー先で**真中ボタン**を押せば良い。コンソール (ターミナル) 等に表示されている文字列をコピーしたいときには、該当文字列のところで左ダブルクリックすれば、ワード単位で選択される。さらにもう一度クリックすれば行が選択される。

第2章　UNIX コマンドの基礎

2.1　使用上の若干の注意

コンソールやターミナル上で、C-s をタイプすると画面表示が固まってキー入力ができなくなることがある。このときには C-q をタイプすれば再びキー入力を受けつけるようになる。(C-s は、本来、高速な画面表示を一時停止するために用意されているキー操作ですが，現在は画面表示が高速になりすぎたため実用的でなくなってしまったキー操作になっています。この一時停止機能は無効になってるターミナルもあります)

2.2　ファイル名の補間

ファイル名やコマンド名を入力する時には、全部タイプしなくても、最初の数文字をいれて Tab キーを押せば、適宜補間される。例えば、以下のように ディレクトリ program などがあるとする。

```
$ ls -F
program/ c/ tex/
```

このとき、

```
$ cd p[ここで Tab を押す]
```

とすれば、p に続いて文字列”program” が補間される。

また、コマンドを実行するときにも同様に最初の数文字を入力して Tab を押せば補間される。候補が複数ある場合には、その候補一覧が表示される。

2.3　コマンドの起動・中断

コマンドを起動するには、コマンド名をコンソール上や kterm 上で入力する。UNIX では同時に複数のプログラムを起動することができる。コマンドの実行を**バックグラウンド処理**で行う (kterm などを占有しないようにする) には、コマンドの後ろに & をつけて実行する

```
$ ./command &
```

emacs 等を起動するときには、これによってバックグラウンド処理にするとよい。もし、& をつけ忘れて立ち上げたときには、コマンドの実行中に **C-z でコマンド中断後、bg とタイプ**すれば、バックグラウンド処理に移行する。また、C-z で中断したりバックグラウンドで走らせているジョブを、ふたたびコンソール上で走らせるには、**fg とタイプ**すればよい。なお、bg は backgroud, fg は foreground の略である。

現在バックグラウンドで走っているジョブは jobs コマンドで確認できる。

```
$ emacs &
$ gcalctool &
$ jobs
[1]- Running    emacs &
[2]+ Running    gcalctool &
```

このように複数のジョブがバックグラウンドで動いているときには、**ジョブ番号**が順につけられる。fg によりフォアグラウンドに切替えるジョブを指定するには、このジョブ番号に%をつけて指定する。

```
$ fg %2
xcalc
```

ジョブ番号を指定しなかった時には、ジョブ番号に'+' がついてるジョブがフォアグラウンドに移される。

暴走してしまった処理中のコマンドを**強制終了**するには、C-c を用いる。他に強制終了を行う方法はいくつかある。2.5.4 節も参照すること。

2.4 コマンド履歴について

実行したコマンドの履歴はしばらく記憶されてるので、C-p や C-n で前に実行したコマンドを探して再実行を簡単にできる。これまでに実行したコマンドの一覧を見るには history コマンドを実行する。

```
$ history
```

これまで実行したコマンドを検索したいときには C-r (前方検索) C-s (後方検索) を使うことが出来る。

2.5 UNIX の基本コマンド (のごく一部)

UNIX 上で頻繁に用いるコマンドの一部を以下に示す。使い方の詳細は man コマンド等で調べること。

2.5.1 基本操作

コマンド	意味
ls [オプション]	現在のディレクトリのファイル一覧を表示する
-l	ファイルの属性等詳しい情報も表示
-a	'.' から始まる隠しファイルも表示
-R	サブディレクトリにあるファイルも再帰的に表示
rm [オプション] <ファイル名>	ファイル削除
-r	指定ディレクトリと、その下にあるファイル全てを再帰的に削除
cp <ソース> <コピー先>	ファイル・ディレクトリのコピー
mv <ソース> <移動先>	ファイル・ディレクトリの移動 (名前変更)

2.5.2 ディレクトリ関連

コマンド	意味
cd ＜ directory 名＞	ディレクトリ移動 (上に行くときは cd ..)
cd -	一つ前にいたディレクトリに移動
mkdir ＜ directory 名＞	ディレクトリをつくる
rmdir ＜ directory 名＞	ディレクトリを消す
pwd	現在いるディレクトリ名を表示

2.5.3 ファイル閲覧・ファイルの情報取得

コマンド	意味
`cat <ファイル名 1> <ファイル名 2> ...`	ファイルをつなげて標準出力に表示
`less <ファイル名>`	ファイルの中身を表示 ('f' or 'Space': 次ページ, 'b': 前ページ, 'q': 終了, 'h': ヘルプ)
`lv <ファイル名>`	less を様々な言語コードに対応したもの
`tail [-<行数>]<ファイル名>`	ファイルの下から 10 行を表示
`-20`	下から 20 行を表示 ('-' に続いて行数を指定できる)
`wc <ファイル名>`	ファイルの文字数、ワード数、行数 を表示
`touch <ファイル名>`	ファイルのタイムスタンプを更新する。もし指定した名前のファイルがなければ、大きさ 0 のファイルをつくる。
`grep [-r] <キーワード> <ファイル名>`	指定ファイルから、キーワードを含む行を検索する。
`-r`	サブディレクトリ以下のファイルも再帰的に検索する。
`sort <ファイル名>`	ファイルを行単位でソート (アルファベット順) した結果を出力する。
`diff <ファイル 1> <ファイル 2>`	ファイル 1 とファイル 2 の違いを表示する

2.5.4 プロセス管理 (ps, top, kill, killall, xkill)

UNIX 上では同時に様々なプログラム (ジョブ) が走っている。現在走っている**ジョブ一覧**を知りたいときには、

```
$ ps auxw
```

でわかる。自分が走らせてるジョブだけ知りたいときには単に `ps` でも良い。(`ps` コマンドのオプションの意味は、`jman ps` で調べること。)

また、コマンド top を使うと、CPU 使用率やメモリ使用率でソートされた結果が表示される。(`top` コマンドの終了は'q', 使い方が分からないときには'h' を押す。)

この、`ps` や `top` の出力を見ると、**プロセス ID(PID)** という項目がある。このように各ジョブにはそれぞれ識別番号がついている。あるジョブが暴走して止まらなくなった時には、そのジョブの PID を調べて、以下を実行すれば、そのジョブの**強制終了**をすることができる。

```
$ kill <PID>
```

これでも停止しないときには、オプション`-9` をつける。

```
$ kill -9 <PID>
```

ジョブの停止は top 画面上からも出来る. PID を確認したら 'k' をタイプし、続いて PID を入力すればよい。

また、コマンド名を指定して `kill` を行うコマンド killall もある。例えば `firefox` という名前のプロセスを全て終了させたい時には次のようにする。

```
$ killall firefox
```

ただし、同じ名前のコマンドがいくつか起動しているときうっかり `killall` を使うと、kill したくないものまで消えてしまうので要注意。

X-windows 上で表示されているあるウィンドウを実行しているプロセスを殺す方法には、xkill を実行して、対象のウィンドウをクリックする方法もある。

2.5.5 ファイル圧縮 (gzip, bzip2)

ハードディスクやフロッピー等の限られた容量内に、多くのファイルを置くためには、あまり使わないファイルは圧縮しておくとよい。UNIX でよく使われる圧縮ツールには gzip や bzip2 がある。圧縮率の高さでは bzip2 が定評があるが、gzip に比べて圧縮に時間がかかるのが欠点である。

コマンド	意味
gzip <ファイル名>	指定ファイルを圧縮する。(<ファイル名>.gz というファイルになる)
gunzip <ファイル名>	圧縮されているファイル (*.gz) をもとに戻す。
bzip2 <ファイル名>	指定ファイルを圧縮する。(<ファイル名>.bz2 というファイルになる)
bunzip2 <ファイル名>	圧縮されているファイル (*.bz2) をもとに戻す。

圧縮されていても、ドキュメントファイルは less コマンドで中を見ることができる。

個々のファイルではなく、ディレクトリごと gzip で圧縮するには以下のようにする。

```
$ tar czvf <ディレクトリ名>.tar.gz <ディレクトリ名>
```

これにより、指定ディレクトリを圧縮した、<ディレクトリ名>.tar.gz という一つの圧縮ファイルができる。

bzip2 を用いる場合には、

```
$ tar czvf <ディレクトリ名>.tar.bz2 <ディレクトリ名>
```

と、出力ファイル名の拡張子を bz2 するだけで良い。

もとに戻すには、以下を実行する。

```
$ tar xzvf <tar.gz ファイル or tar.bz2 ファイル>
```

また、もとに戻さず単に tar.gz ファイルの中にどのようなファイルが含まれているか知りたいときには、

```
$ tar tzvf <tar.gz ファイル or tar.bz2 ファイル>
```

とする。

なお、Vine Linux では、ディレクトリの圧縮のために、gzipdir, bzip2dir というコマンドも用意されている。

```
$ gzipdir <ディレクトリ名>
$ bzip2dir <ディレクトリ名>
```

2.5.6 ファイルを探す (which,locate)

あるコマンドを実行するとき、そのコマンドがどこのパスから呼び出されているかを知るには which を使う。

```
$ which less
/usr/bin/less
```

ある文字列を含むファイルがどこのパスにあるか、その一覧を知りたいときには locate を使う。

```
$ locate stdio.h
/usr/lib/bcc/include/stdio.h
/usr/lib/perl5/5.8.6/i386-linux-thread-multi/CORE/nostdio.h
/usr/include/bits/stdio.h
/usr/include/isc/stdio.h
/usr/include/glib-2.0/glib/gstdio.h
/usr/include/stdio.h
```

2.5.7 文字コードの変換・判定 (nkf)

日本語の文字コード (日本語の各文字を表す数値) には JIS, SJIS, EUC,UTF-8 など様々な企画がある。そのため，例えば MS Windows で作成したファイルを Mac や Linux で見ようとすると文字化けする事がある。そのような時には nkf コマンドである文書の文字コードが何かを判定したり，文字コードの変換を行ったりすることができる。

コマンド	意味
nkf [オプション] ファイル名	文字コードの判定, 変換
-g	文字コードの判定
-j	JIS(ISO-2022-jp) コードに変換
-s	SJIS コードに変換
-e	EUC コードに変換
-w	UTF-8 コードに変換

コマンド nkf には他にもいろいろな変換機能があるが，インストールされているバージョン等により若干異なるので `man nkf` で一度確認しておくと便利。

2.5.8 ネットワーク関連のコマンド

コマンド	意味
ping <IP アドレス or ホスト名>	指定したホストとネットワークで接続されているかを確認
ssh <IP アドレス or ホスト名>	指定したホストにログイン (通信は暗号化)
ssh <ユーザ名>@<IP アドレス or ホスト名>	指定したホストに指定ユーザ名でログイン (通信は暗号化)
scp <IP アドレス or ホスト名>:<ファイル 1> <ファイル名 2>	指定したホストのファイルを、指定ファイル名でコピー (通信は暗号化)

ping

あるマシンがネットワークにつながっているかどうかを確認するには、ping コマンドを用いる。

```
$ ping venus.hogehoge.ac.jp
PING venus.hogehoge.ac.jp (133.62.236.100) from 133.62.236.98 : 56(84) bytes of data.
64 bytes from venus.hogehoge.ac.jp (133.62.236.100): icmp_seq=0 ttl=255 time=0.4 ms
64 bytes from venus.hogehoge.ac.jp (133.62.236.100):
icmp_seq=1 ttl=255 time=0.3 ms

-- venus.hogehoge.ac.jp ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.3/0.3/0.4 ms
```

上の例では、ネットワークを介して venus.hogehoge.ac.jp に到達可能であることがわかる。

ネットワーク接続 (ssh,scp)

ネットワークにつながっているマシンにログインして、さまざまな操作を行うには、暗号化通信を行える ssh がよく使われる。

以下の例は ssh で venus.hogehoge.ac.jp に接続した例である。

```
$ ssh jun@venus
Host key not found from the list of known hosts.
Are you sure you want to continue connecting (yes/no)? yes
```

このように、あるホストに始めて ssh で接続するときには、”そのホストには接続したことがないけど、本当に接続しますか?” と質問があるが、”yes” と答えると以下のようにログインを行える。

```
Host 'venus' added to the list of known hosts.
jun@venus's password:
Last login: Thu Aug 10 11:47:30 2000 from muse.hogehoge.ac.jp
```

リモートホストにあるファイルを自分が操作しているローカルホスト上にコピーしたい時には、scp を用いる。

```
$ scp venus:doc/memo .
```

上の例は、venus 上の ~/doc/memo を手元にコピーするためのコマンドである。指定ファイル名には、ワ

イルドカード (後述) も使える。

2.5.9 いろいろな情報を得る

コマンド	意味
man < コマンド >	システムにインストールされてるコマンドの説明表示
df	ディスク使用量を知る。

2.6 ワイルドカードについて

'*' は**ワイルドカード**と呼ばれ, 任意の文字列をさす。

```
$ ls a*
```

とすれば、a ではじまるファイル名の一覧表示.

```
$ ls *a*
```

とすれば、a を含むファイル名の一覧表示.

```
$ cat *.c
```

とすれば、すべての c プログラムの中身表示

'?' は任意の一文字をさす。

```
$ ls c?
```

とすれば、ファイル名が “c+1 文字” であるようなファイルが (あれば) 表示される。

2.7 パイプについて

'|' を**パイプ**とよぶ。パイプの左側で実行したコマンドの標準出力が、パイプの右側のコマンドの標準入力になる。

例えば以下のように、ls のファイル一覧出力を lv (less でも良い) を使って見ることができる。たくさんファイルがあるとき便利である。

```
$ ls -l | lv
```

ls の出力を sort コマンドでソートして、さらに less で見たいときには以下のようにする。(本当は ls の出力はすでにソートされているので sort コマンドを使う必要は無いのだが...)

```
$ ls | sort | lv
```

2.8 リダイレクトについて

'>' を**リダイレクト**よぶ。リダイレクトの左で実行したコマンドの標準出力が、リダイレクトの右に書いたファイルに出力される。

例 1: ls の標準出力をファイル filelist に書き込む。

```
$ ls > filelist
```

例 2: ファイル data をソートした結果を data2 に書き込む。

```
$ sort data > data2
```

2.9 シンボリックリンク

~/tex/というディレクトリにいるときに、~/c/result というデータファイルを頻繁に参照する必要があるとしよう。このとき、~/c/result を毎回参照するのは面倒なので、~/tex/にコピーするのも一手だがこれだと、~/c/result を修正したときに毎回コピーし直さないといけない。なによりもディスク消費も増える。このような時には以下のように**シンボリックリンク**を作る。

```
$ cd ~/tex
$ ln -s ../c/result result
```

このあと、ls -l ~/tex を実行してみよう。以下のようなファイルが出来ている。

```
lrwxrwxrwx   1 jun      users          11 Mar 24 14:28 result -> ../c/result
```

これで、~/tex/result を参照すると、~/c/result が参照される。このとき、

~/tex/result から~/c/result に**シンボリックリンク**を張っている

という。

シンボリックリンクはディレクトリに対しても作れる。

```
$ cd ~/tex
$ ln -s ../c c
```

とすれば、~/tex/c を見ると~/c が参照される。

2.10 システムの停止

UNIX を動かしているシステムを停止したいとき、いきなり電源を落してはいけない。メニュー選択によって終了するか、以下のコマンドを用いる。

```
$ /usr/sbin/shutdown -h now
```

停止後すぐに再起動したい時 (reboot) には、以下のように-r オプションを使う。

```
$ /usr/sbin/shutdown -r now
```

第3章　シェル(Bash)

UNIX では、入力した様々なコマンドの解釈・実行を行うためのユーザインタフェースとして、**シェル**が起動される。シェルには bash, tcsh, zsh ほか様々な種類があり、それぞれ tab キーによるコマンドの補間機能や、プログラミング能力その他若干の違いがある。Linux では多くの場合 bash が標準なので、以下では bash について説明する。

3.1　設定ファイル

ログイン時に一度だけ実行する命令は、~/.bash_profile に記述する。また、シェルの起動時に実行するコマンドは、通常~/.bashrc に記述する。(~/.bashrc の中身はログイン時や kterm の起動時に毎回実行される)

3.2　エイリアス

長いコマンドを毎回うつのが面倒なとき、**エイリアス** (別名) を定義しておくことができる。たとえば、以下の記述を~/.bashrc に加えておくと、em とタイプすれば emacs が起動するようになる。

```
alias em='emacs'
```

~/.bashrc に記述を加えたあとは、新たな設定を有効にするために、

```
$ source ~/.bashrc
```

を実行して~/.bashrc を読み込むか、ログインをしなおすこと。

~/.bashrc を見ればわかるように、すでにいつかのエイリアスが定義されている。例えば、

```
alias ls='ls -F --color=auto'
```

という定義は ls を実行すると、ls -F --color=auto が実行されることを意味する。

もし、上のような ls のエイリアスを無効にして実行したい時には、\をコマンドにつけて、

```
$ \ls
```

を実行する。

現在 ls にどのようにエイリアスが設定されているかは alias コマンドで知ることができる。

```
$ alias ls
```

現在のエイリアスの設定を一時的に無効にするには unalias を使う。

```
$ unalias ls
```

ls のエイリアスは継続的に無効にしたい時には、当然.bashrc のエイリアス設定も消去する必要がある。

あるコマンドがどのようなパスから、もしくはエイリアスから実行されているかを知りたいときには which を使う。

```
$ which rm
alias rm='del'
        /usr/bin/del
$ unalias rm
$ which rm
/bin/rm
```

上記の例では，rm は del のエイリアスになっており，del で/usr/bin/del が起動されていることがわかる。しかし、このエイリアスを消去すると/bin/rm が起動される。

3.3 パス

なんらかのコマンドをタイプして実行しようとすると、環境変数 PATH(パス) で設定されているディレクトリから該当名のコマンドが捜索され、実行される。現在の PATH 設定は、

```
printenv PATH
```

で参照できる。(単に printenv とタイプすると、設定されてる全ての環境変数が表示される)

例えば、~/bin というディレクトリに入ってるコマンドも、PATH に追加したいときには以下を実行する。

1. ~/.bashrc に以下の行を追加

   ```
   PATH=$PATH:$HOME/bin
   export PATH
   ```

   HOME は各ユーザのホームディレクトリを示す変数である。printenv HOME を実行すると、$HOME がなにかわかる。

2. 設定 (~/.bashrc) を読み込む。

   ```
   $ source ~/.bashrc
   ```

3.4 シェルスクリプト

3.4.1 シェルスクリプトの基本

エイリアスは、長くても一行程度ですむようなコマンドの別名を定義するのに使う。さらに長い一連の命令をひとつのコマンドとするには、通常**シェルスクリプト**と呼ばれるファイルを作成する。

例えば、以下のような内容のファイルを ls.sh という名前にして作ってみよう。

```
#!/bin/bash

# まず現在のディレクトリを調べる。
echo -n "現在のディレクトリは "
pwd

# どんなファイルがあるかを表示する。
echo "以下のファイルが見付かりました。"
ls
```

次に、以下を実行する。

```
$ chmod +x ls.sh
```

これは、ls.sh を実行可能なファイルにする命令である。(実行許可を取り消すには`$ chmod -x ls.sh` とする。) ここで `~/ls.sh` を実行すれば、現在のディレクトリ名と、中にあるファイル一覧が表示される (echo は 文等を表示する命令で、`-n` は表示後に改行しないためのオプション)。このように、実行したい命令をずらずらと書いたファイルをつくり、ファイルの先頭に`#!/bin/bash` という行をいれれば、記述した命令を順次実行できるシェルスクリプトになる。

また、各行で記号`#`があるとき、それ以降の文字は**コメント文**とみなされて無視される。

3.4.2 シェルスクリプトの引数

シェルスクリプトの各引数は`$0,$1,...` で参照できる。引数の数は`$#`で参照できる。

```
#!/bin/bash

echo "これが第 0 引数" $0
echo "これが第 1 引数" $1
echo "これが全ての引数" $@
echo "これが引数の数" $#
```

3.4.3 文字変数

Bash スクリプトで**文字変数**を定義するには'=' を使って定義すればよい。以下の例では、変数 FNAME に `test.c` が、変数 ARG に引数`$1` が変数 BINDIR に`/usr/bin` が代入される。

```
FNAME="test.c"
ARG="$1"
BINDIR="/usr/bin"
```

ここで、= **の両側にはスペースが入らないことに注意!** 変数の値を参照するには、変数名に`$`をつける。また、変数名を`{}`で囲むことが多い。(必ずしも囲む必要はないが、変数名の範囲を明確にできるのでトラブルがおきにくい)

以下は、上のように定義した変数を参照する例である。

```
echo "FNAME は" ${FNAME} ", ARG は" ${ARG} "です"
ls ${BINDIR}
```

echo で表示する変数は、ちょっと手を抜いて""の中に入れてしまっても良い。

```
echo "FNAME は${FNAME}, ARG は${ARG}です"
ls ${BINDIR}
```

3.4.4 変数の操作

Bash スクリプトでは変数の値を簡単に操作できる機能がある。以下ではその一部を紹介する。

命令	意味
${VAL%word}	変数${VAL}の値の後ろから word に合致する最小部分を削除した値
${VAL%%word}	変数${VAL}の値の後ろから word に合致する最長部分を削除した値
${VAL#word}	変数${VAL}の値の頭から word に合致する最小部分を削除した値
${VAL##word}	変数${VAL}の値の頭から word に合致する最長部分を削除した値

以下はシェルスクリプトでの使用例である。

```
#!/bin/sh
DIR=/usr/local/bin
echo ${DIR%/*}
echo ${DIR%%/*}
echo ${DIR#/*/}
echo ${DIR##/*/}
```

実行結果は以下の通り

```
/usr/local

local/bin
bin
```

3.4.5 シェルスクリプトの終了

シェルスクリプトは exit が実行された時終了する。正常終了の時には `exit 0`、異常終了の時には `exit 1` (0 以外を指定) とする場合が多い。

3.4.6 if 文

Bash スクリプトでの if 構文は以下の通り。

```
if [ 条件文 ] ; then
...
else
...
fi
```

以下に条件文の一部を示す。

条件文	意味
-d <文字>	<文字>の名前のディレクトリがある時真
-f <文字>	<文字>の名前のファイルがある時真
<文字 1> = <文字 2>	<文字 1>と<文字 2>が等しい時真 ('=' **の両側にスペースがあること注意**)
<文字 1> ! = <文字 2>	<文字 1>と<文字 2>が異なる時真
! <条件文>	<条件文>が偽であるときに真 (NOT)

3.4.7 for 文

Bash スクリプトでの for 構文は以下の通り。

```
for <変数> in <値 1> <値 2> ....; do
...
done
```

<変数>に <値 1> <値 2> .... が順に代入され、do と done で囲んだ部分が繰り返し実行される。

以下はファイル a, b, c をそれぞれソートして a2, b2, c2 にするスクリプトである。

```
for i in a b c; do
   sort $i > ${i}2
done
```

in 以下には値を並べるかわりに、あるコマンドの出力を用いることもできる。以下は `ls *.dat` で表示されるファイルについて、先の例と同様の処理を行うスクリプトである。(コマンドは ‘ で囲むこと)

```
for i in ‘ls *.dat‘; do
   sort $i > ${i}2
done
```

以下は現在のディレクトリの下にあるディレクトリの一覧を表示するスクリプトである。

```
for i in ‘ls‘; do
   if [ -d $i ]; then
      echo $i
   fi
done
```

3.4.8 case 文

ある変数の値に応じて様々な処理を分岐させる時には、if 文を使うよりも case 文を使う方が便利なことが多い。

```
case <変数> in
        <値 1>) 文 1
                ;;
        <値 2>) 文 2
                ;;
        *) 文 (default)
                ;;
esac
```

変数が値 1 の時は文 1 が、値 2 の時は文 2 が、いずれの値にも該当しないときには、文 (default) が実行される。各文の終りには ;; を記述すること。;; が読み込まれると、case 文は終了する。

以下は、引数で与えたファイルの拡張子に応じて解凍を行う例である。(“|” は OR を表す。)

```
case "$1" in
        *.tar.gz|*.tar.bz2) tar xzvf $1  ;;
        *.gz) gunzip $1 ;;
        *.bz2) bunzip2 $1 ;;
esac
```

3.4.9 関数

シェルスクリプトでも C 言語などのように関数を定義できる。関数の宣言は以下のフォーマットになる。

```
  関数名 (){
     実行内容
  }
```

以下は簡単なシェルスクリプト例。引数の数が不適切なときに、関数 Usage() を呼出し、実行コマンド名からディレトリ名を削除したものを引数として渡している。関数に渡された引数も$0,$1,... でアクセスできる。

```
#!/bin/sh

Usage(){
        echo "Usage: $1 <filename>"
        exit 1
}
if [ $# -ne 1 ]; then
        Usage ${0##/}
fi
echo $1
exit 0
```

第4章　エディタ(emacs)

4.1　UNIX上で動くエディタ

プログラム等をつくるときに、UNIX上でよく使われてきたエディタにemacsがある。最近はVS Codeや Sublime Textなども人気がある。いろいろ試して気に入ったものを使うと良い。

以下ではemacsの使い方を概説する。ちなみに，`emacs`と同様の操作およびキー操作のエディタで軽いものにng, jedなどもある。`emacs`はコンソール上ではオプション`-nw`をつけて起動する。X Window上で立ち上げるときにはオプションは不要。

ファイル操作等はメニューからも行うことができるが、できるだけキー操作を覚えると作業効率がよくなるので、できるだけキー操作を覚えることを勧める。`emacs`では日本語チュートリアルがある (メニューのhelpから呼べる) ので、これを利用してみるとよい。

いろいろなキーを押してるうちに表示がおかしくなったら

```
C-g
```

を2回続けてタイプすると、もとの状態に (大抵) 戻る。

4.2　注意事項

emacsを同時に複数起動してはいけない。編集用に複数のウィンドウが必要な時には、一つのemacsから新しいウィンドウを開けられる。(4.5参照)。このほうが、操作もいろいろ便利で、メモリ使用量もはるかに少なくてすむ。

4.3　基本操作

キー操作	意味
C-{xf}	ファイルを開く
C-{xs}	ファイルを保存
C-{xc}	終了
C-{g}	エラーがあったときとりあえず何度か押してみる
C-x i	ファイルをマウスカーソルの位置に挿入
C-x k	現在編集中の文書を破棄する
C-x u	アンドゥ(実行したコマンドの取消)
C-s	文字検索 (カーソル行以降で検索)
C-r	文字検索 (カーソル行より前で検索)
M-x query-replace	文字置換 (確認あり)
M-x replacestring	文字置換 (確認なし)

4.4 移動

キー操作	意味
C-v	次の画面に進む
M-v	前の画面に戻る
C-b	一文字左へ
C-f	一文字右へ
C-p	一文字上へ
C-n	一文字下へ
C-d	カーソル位置の文字を削除
C-k	カーソル位置から行末までの文字を削除
C-e	行の一番右へ
C-a	行の一番左へ
M-f	一単語右へ
M-b	一単語左へ
M-g <行番号>	指定行へ移動 (emacs)
M-x goto-line	指定行へ移動

4.5 画面操作

キー操作	意味
C-x 2	画面を上下に分割
C-x o	分割した上下の画面間を移動
C-x 0	分割した画面のうちカーソルのあるほうを閉じる
C-x 1	分割した画面のうちカーソルの無いほうを閉じる
C-x 5 2	もう一つウィンドウを開く
C-x 5 o	ウィンドウ間で移動
C-x 5 0	カーソルのあるウィンドウを閉じる
C-x b	読み込んである他のファイルを表示 (名前を指定)

4.6 ファイル一覧窓での操作

C-{xb} で編集中のファイル一覧が表示される。この一覧表示をしてる窓に C-x o で移動すると、各ファイルについていろいな操作を行える。今回は、そのファイルの表示に関するコマンドのみ紹介する。

キー操作	意味
C-{xb}	現在読み込んでるファイルの一覧表示
1	カーソル位置のファイルを現在のウィンドウいっぱいに表示する
2	カーソル位置のファイルを現在の窓に表示する
n	カーソルと次の行へ進める
p	カーソルと前の行へ戻す

4.7 カット/コピー/ペースト

キー操作	意味
C-space	始点のマーク
M-w	始点から現在のカーソル位置までを記憶
C-w	始点から現在のカーソル位置までを削除して記憶
C-y	記憶内容をカーソル位置にペースト (出力)

編集中のファイルの**一部分を別の場所にコピー**するには以下のように行う。

1. コピーしたい部分の先頭にカーソルを移動する
2. C-space をタイプ (これで先頭位置が記憶される)
3. コピーしたい部分の終りにカーソルを移動する
4. M-w を押す (これで先頭位置からこの終りの部分までが記憶される。この部分を**リージョン** (region: 領域) と呼ぶ)
5. コピー先にカーソルを移動する
6. C-y をタイプ。これでコピー完了。

一部分を削除したい時には、上のコピーの手続きで、M-w のかわりに、C-w をタイプすれば、設定したリージョンが削除・記憶される。

一部分を移動したい時には、上の削除を行った後、移動先へカーソルを持って行きコピーの場合と同様に C-y をタイプすれば、削除された領域がそこにペーストされる。

4.8 C/C++のプログラムを書く

emacs で拡張子が .c や .cc といったファイルを読み込むと、自動的に c/c++ プログラムの編集用モードに切り替わる。メニューには “C” もしくは”C++” といったメニューができる。

C-{cc} を押せばすぐにコンパイルを行うことができる。

キー操作	意味
C-{cc}	コンパイル実行
C-c c	リージョンをコメントする
C-{uc} c	リージョンをアンコメントする

4.9 Emacs のカスタマイズ

emacs は, 設定ファイル.emacs を編集することで，いろいろな機能を追加できる。例えば，以下を.emacs に追加すると，一行が 80 字以上になった時には自動改行できる。

```
(setq fill-column 80)
(setq text-mode-hook 'turn-on-auto-fill)
(setq default-major-mode 'text-mode)
```

google 先生に「emacs カスタマイズ」でお伺いしてみると，便利な機能をいろいろ発見できる。

4.10 LaTeX 文書の作成 (auctex モード)*

LaTeX は数式を含む文書作成に優れた文書整形システムである。研究室では、emacs 上に TeX ファイルを読み込むと auctex モードになり、LaTeX 文書の作成が容易になるように設定してある。(Vine Linux のデフォルトの設定では yatex が起動する)

4.10.1 auctex の使いかた

auctex の操作

このモードの時には、emacs 上部に “LaTeX”, “Headings”, “Show”, “Hide” というメニューが出て来るので、これをクリックすればいろいろな機能を見付けることができる。以下には代表のものだけ紹介する。

4.10.2 TeX 文書の処理

キー操作	意味
C-{cc}	TeX ファイルの処理実行 (LaTeX2e, View, Print など指定)
C-c ‘	platex コマンド等の処理がエラーを出したとき、そのエラー箇所を表示
C-{cl}	platex コマンド等の処理中、その処理の様子を表示

4.10.3 TeX 文書の作成

キー操作	意味
C-{ce}	LaTeX コマンド挿入 (\begin{},\end{}などの挿入)
C-{uce}	マウスカーソルの位置が囲まれている LaTeX コマンドの変更
M-Return	\item 挿入
C-c ;	リージョンをコメント
C-c :	リージョンをアンコメント

4.10.4 TeX 文書の表題表示

長い文章つくるときに、セクション見出しのみを表示したりできます。

キー操作	意味
C-c @ C-t	折り畳む (セクション一覧)
C-c @ C-c	折り畳む (マウスポインタのあるセクションのみ)
C-c @ C-a	広げる (全文表示)
C-c @ C-e	広げる (マウスポインタのあるセクションのみ)

第5章　ソフトウェアのインストール方法 (apt/rpm)

5.1　パッケージ管理について

Ubuntu 系 Linux (Ubuntu/Mint 等) では，アプリケーションを deb 形式と呼ばれるパッケージ形式で配布している。パッケージ管理 (ダウンロードやインストール，削除など) には apt コマンドを使う。RedHat 系 Linux (CentOS/Fedora/Vine 等) では，rpm で配布されており，パッケージ管理には CentOS や Fedora では yum を，Vine Linux では apt を使う。

以下では，apt や rpm の利用方法を簡単に説明する。

注意) パッケージのインストールや削除等は管理者権限で行う必要がある。関連コマンドの実行時は‘sudo‘を利用すること。

5.2　apt を利用したパッケージのインストール・一斉更新

apt を利用すると、あるディレクトリやインターネット上の各サイトにある rpm パッケージのダウンロードとインストール、アップグレードを簡単にできる。パッケージ依存性も同時にチェックして、必要なものは同時にインストールしてくれる。パッケージ入手先を追加・変更したいときは，必要に応じて /etc/apt/sources.list に登録・修正する。

1. まず必ず行うおまじない: すでにインストールされているパッケージ情報と、sources.list に登録されているサイトにある最新パッケージ情報を入手する。

```
# apt update
```

2. 特定のパッケージをインストールするとき

```
apt install package [package ...]
```

これで、指定した package の入手・インストールが自動的にされる。

3. 既にシステムにインストールされているパッケージを、最新版に更新したいとき

```
apt upgrade
```

このとき、同時に必要になる追加パッケージがあれば、いっしょにインストールしてくれる。ただし、更新によって、既にインストールされているパッケージが、同様の機能を持つ別のパッケージに置き換えられるものや、依存性の関係で削除されるパッケージものがある場合には、更新は行われない。

4. 既にインストールされているパッケージを、パッケージの置き換えや削除も含めて最新版に更新したいとき

```
apt dist-upgrade
```

現在インストールされているパッケージを、最新のパッケージにする点は apt upgrade と同じだが、同じ機能をもつ別のパッケージへの差し替えや、依存性に問題の生じるパッケージの削除も行われる。便利だけど、パッケージの削除が行われるときには注意が必要。

5.2.1 パッケージの削除

```
apt remove package [package ...]
```

5.2.2 パッケージの情報取得

1. まず行うおまじない

```
apt-cache gencaches
```

このコマンドで、パッケージの最新情報を取得できる。

2. パッケージの情報取得

```
apt-cache show package [package ...]
```

このコマンドで、指定したパッケージのバージョン、機能、ライブラリやパッケージの依存関係等が表示される。もっと詳しい情報を知りたいときには以下を実行する。

```
apt-cache showpkg package [package ...]
```

5.2.3 パッケージとファイルの関係についての情報取得

1. 準備

```
apt install apt-file
apt-file update
```

2. 指定したファイルをインストールしたパッケージ名を表示する。

```
apt-file search <ファイル名>
```

3. 指定したパッケージに含まれるファイル一覧を表示する。

```
apt-file list <パッケージ名>
```

5.3 rpm を利用したパッケージ管理

例えば，skype-0.93.0.3-fc2.i386.rpm というパッケージを入手したとする。このファイル名のうち skype はパッケージ名、0.93.0.3 はバージョン番号 (ソフトウェアのバージョン)、fc3 がリリース番号 (rpm パッケージのバージョン) を指す。

- rpm パッケージに関する情報をみる

```
# rpm -qip skype-0.93.0.3-fc2.i386.rpm
ame        : skype                         Relocations: (not relocatable)
Version     : 0.93.0.3                           Vendor: (none)
Release     : fc2                           Build Date: 2004?12?22? 00?25?23?
Install Date: (not installed)               Build Host: localhost.localdomain
Group       : Internet                      Source RPM: skype-0.93.0.3-fc2.src.rpm
Size        : 5561226                       License: Commercial
Signature   : (none)
Summary     : Skype is free Internet telephony that just works
Description :
Skype offers free superior sound quality Internet telephony. In addition, it
includes:
...
```

上記コマンドオプションの q は query(問合せ), i は information(情報), p は package(パッケージ名) を意味する。

- rpm パッケージに含まれるファイルの一覧をみる

```
# rpm -qlp skype-0.93.0.3-fc2.i386.rpm
/usr/bin/skype
/usr/share/applications/skype.desktop
/usr/share/pixmaps/skype.png
....
```

上記コマンドオプションの l は list(一覧) を意味する。

- rpm パッケージのインストール

```
# rpm -ivh skype-0.93.0.3-fc2.i386.rpm
```

上記コマンドオプションの i は install(インストール), v は verbose(言葉数の多い, (コマンド実行中に詳しい情報を表示)), h は hash(# 印をインストール中に表示) を意味する。

- インストールしたパッケージのファイル一覧をみる

```
# rpm -ql skype
/usr/bin/skype
/usr/share/applications/skype.desktop
/usr/share/pixmaps/skype.png
....
```

- システムにインストールしてある rpm パッケージの一覧を表示

```
# rpm -qa
```

上記コマンドオプションの a は all を意味する。

- インストールしてあるパッケージのバージョンをみる

```
# rpm -q skype
skype-0.93.0.3-fc2
```

- インストールしてあるパッケージの情報を得る

```
# rpm -qi skype
ame         : skype                         Relocations: (not relocatable)
Version     : 0.93.0.3                            Vendor: (none)
Release     : fc2                           Build Date: 2004?12?22? 00?25?23?
Install Date: (not installed)               Build Host: localhost.localdomain
......
```

- rpm パッケージのアップグレード

```
# rpm -Uvh skype-1.3.0.53-fc5.i586.rpm
```

- インストールしてある rpm パッケージの削除

```
# rpm -e skype
```

上記コマンドオプションの e は erase(削除) を意味する。

- システム上にあるファイルがなんというパッケージのものかを知る

```
$ rpm -qf /usr/bin/less
less-382-0vl4
```

上記コマンドオプションの f は file(ファイル名) を意味する。

第6章　システム管理**

6.1　システム環境の設定

アクセス制限 (ファイアウォール)、利用するキーボードやマウスの種類の変更、サウンドカードの設定は root 権限で setup コマンドを使えば設定できる。

```
# /usr/sbin/setup
```

GUI ベースの設定ツール webmin も様々なシステム設定をするのに便利。`apt-get install webmin` でインストールし、ブラウザで `https://localhost:10000/` にアクセスすれば利用できる。

6.2　NFS(Network File System)

研究室では venus のハードディスクに各ユーザのホームディレクトリを用意し、どの PC にログインしても venus のホームを利用するように設定してある。venus のようにディスクスペースを提供するマシンを**ファイルサーバ**もしくは、**NFS**(Network File System) **サーバ**と呼ぶ。

ここでは、ファイルサーバの設定と、ファイルサーバを利用する各クライアントの設定を説明する。

6.2.1　NFS サーバの設定

ファイルサーバ hoge.org のディレクトリ`/home` を共有したいときには、共有許可を出すための設定ファイル/etc/exports に次のように書き込む。

```
/home   192.168.0.2(rw), *.fuga.org(ro)
```

この例では、IP アドレスが 192.168.0.2 のマシンに対して読み書き両方の許可 (rw) を出し、fuga.org というドメイン名をもつマシンに対して読込み許可 (ro) を出している。

`/etc/exports` の編集を行ったら、この設定を有効にするため exportfs コマンドを実行する。

```
# /usr/sbin/exportfs -a
```

(exportfs のオプションについては man exportfs 参照)

最後に NFS サーバデーモンが起動しているかを確認する。

```
# /etc/rc.d/init.d/nfs status
```

停止している場合には起動を行う。

```
# /etc/rc.d/init.d/nfs start
```

6.2.2 NFS クライアントの設定

ファイルサーバ hoge.org の/home を、クライアント上で/home/hoge として利用するには、以下のコマンドを実行する。

```
# mount -t nfs hoge.org:/home /home/hoge
```

この作業を 「hoge.org のホームをマウントする」と言う。これで/home/hoge をローカルディスクと同じ様に使える。もし、うまくいかなければ、mount 実行のエラーメッセージや hoge.org の/var/log/messages に出るエラーメッセージが修正のヒントになる。

6.3 autofs を利用したマウント設定

研究室では、autofs を使ってどの PC からログインしてもファイルサーバ (venus) にあるホームを自動でマウントするように設定してある。autofs を使うと、特定のディレクトリにアクセスするだけで、自動的に mount を行い、一定時間アクセスすると自動で umount を行ってくれるので、大変便利。

ここでは、/misc/cd にアクセスがあったときに、

```
$ mount -t iso9660 /dev/cdrom /misc/cd
```

を自動で実行して、/misc/dvd にアクセスがあったときには、

```
$ mount /dev/sda /mnt/dvd
```

を実行するように設定する。

まず autofs の設定ファイル/etc/autofs.master には次のように書く。

```
/misc   /etc/auto.misc  --timeout=60
```

これは、「ディレクトリ/misc にアクセスがあったときにはファイル/etc/auto.misc を参照しなさい」という意味。最後の--timeout=60 は、/misc へのアクセスが 60 秒無いときには、umount するためのオプション設定。

次にファイル/etc/auto.misc を次のように編集する。

```
cdrom           -fstype=iso9660,ro,users :/dev/cdrom
dvd             -fstype=ext2,users       :/dev/scd0
```

一行目の第一フィールド (cdrom) は、/misc/cdrom にアクセスがあったとき、そこに第三フィールドのデバイスを mount することを意味する。第二フィールドの-fstype=からはじまる部分は、mount 時のオプションであり、/etc/fstab に書くマウントオプションと同じ。

設定を行ったら、ディレクトリ/misc を作成し、autofs デーモンを起動する。このとき、/misc/cdrom や/misc/dvd は作る必要は無い。

```
# mkdir /misc
# /etc/rc.d/init.d/autofs start
```

PC の起動時に autofs を自動で立ち上げるには chkconfig で設定する。

autofs を使って NFS マウントを行うこともできる。例えば、次のように設定すれば、/nfs/hoge/以下にアクセスすると NFS サーバ hoge.org の/home/が自動的に/nfs/hoge にマウントされる。

```
# /etc/auto.master 修正版
/misc   /etc/auto.misc  --timeout=60
/nfs   /etc/auto.home  --timeout=60
/home
```

```
# /etc/auto.nfs の中身
hoge      hoge.org:/home
```

もし、NFS サーバ hoge.org の/home を、クライアント上で/home/hoge にマウントしたいときには、上記の作業の後、シンボリックリンクを張れば OK。

```
# ln -s /nfs/hoge /home/hoge
```

6.4 ちょっとしたセキュリティ向上

LAN に接続されている計算機には、世界中からのアクセスが可能なので、対策を練っておかないど簡単に不正アクセスされてしまう。パスワードが盗まれたりしなくても、怪しげなネットワーク資源やデータの置き場所にされたり、他のサイトを攻撃する中継地点にされてしまうこともある。

ここでは、簡単にできるセキュリティ強化の基本を述べる。

6.4.1 外部からアクセスしたいとき

暗号化通信を行える ssh を使う。暗号化通信をできない telnet や rsh によるアクセスはできないように設定し、また可能であればアクセスを許すサイトも限定する。アクセス制限は setup コマンドの「ファイアウォール設定」でできるが、webmin も便利。

6.4.2 余計なサービス (デーモン) 停止

起動時に自動で起動するデーモンは/usr/sbin/ntsysv や、/sbin/chkconfig --list で確認することができます (初心者にや ntsysv のほうがおすすめ)。使わないサービスは極力起動しないようにしましょう。不要なサービスは、PC の負荷を大きくするだけでなく、セキュリティホールがあれば侵入を許したり悪用されたりする原因となります。

6.5 各ユーザにディレクトリ利用制限 (quota) をつくる

特定のユーザが自分のホームにたくさんのファイルをおくと、その分他の人が使えるディスク容量が減ってしまいます。みんながハードディスクを無駄に消費すると、バックアップのコストが高くなる。不慣れな操作により無駄に大きなファイルを作ってしまい、それに気づかずにいることもある。

このようなことを防ぐためには各ユーザに対して**ディレクトリ使用制限** (quota) を設けると便利。以下では/home としてマウントしている/dev/hda3 に使用制限をつける場合を説明する。ユーザグループに対して、制限をつけることもできる。

6.5.1 quota をかけるパーティションの設定

まず/etc/fstab の編集をする。ユーザに対して quota をかけるパーティションには、オプションに usrquota を指定する。ユーザグループに quota をかけるときには grpquota を指定する。

```
/dev/hda3   /home    ext2    defaults,usrquota,grpquota        1 2
                                    ~~~~~~~~~~~~~~~~~
```

編集後は、システムを再起動するか、以下のコマンドにより quota を有効にする。

```
# quotaon -a              (全ての quota 設定を有効にする)
# quotaon /dev/hda3       (特定の quota 設定を有効にする)
```

6.5.2 ユーザに quota をかける

setquota コマンドで quota を設定する。quota を設定するユーザ名を funya とすると

```
# /usr/sbin/setquota funya /dev/hda3 10000 15000 0 0
# setquota ユーザ名  <パーティション名> <softlimit> <hardlimit> <i-node soft limit> <i-node hard limit>
```

各ユーザの quota は、setquota を使わずに、quota 編集コマンド edquota で設定することも出来る。

```
# /usr/sbin/edquota funya
                    ユーザ名
```

6.5.3 quota 状況をみる

repquota コマンドで、現在の quota の設定情報を知ることができる。

```
# /usr/sbin/repquota -a   (全てのファイルシステムの quota 情報を知りたいとき)
# /usr/sbin/repquota /dev/hda3   (特定のファイルシステムの quota 情報を知りたいとき)
```

quota をやめる

quota の終了は quotaoff コマンドで行える。

```
# quotaoff -a              (全ての quota 設定を無効にする)
o# quotaoff /dev/hda3       (特定のパーティションに対する quota 設定を無効にする)
```

設定を完全に無効にするには /etc/fstab の quota 設定の削除も忘れないようにすること。

第7章　トラブル!!!*

7.1　ログインできない!!!!!

研究室では NIS(Network Information Service) というシステムを用いてユーザパスワードを一元管理している。このようにパスワード等を NIS で共有している環境でログインできなくなる時には、大抵ネットワークのトラブル等があって、パスワード情報の共有が出来なくなっていることが多い。原因を確認するは以下を順に試す。

1. root でログインする (時間がかかることがある)
2. 以下を実行 (NIS クライアントプログラムの実行)

```
$ cd /etc/rc.d/init.d
$ ./ypbind stop
$ ./ypbind start
```

3. このあと、

   ```
   $ ypwhich
   ```

   を実行して、NIS サーバの名前が出てくればログインできるはず。このときは、ntsysv コマンドで、ypbind が起動時に実行されるように設定を行って作業終了。
4. ログインできなければ、他のマシン (NIS サーバ以外) からログインできるかを試す。できるならば、ネットワークの設定がおかしくないか、次節にしたがって確認する。
5. NIS サーバ以外のどのマシンでもログインできなければ、NIS サーバのネットワーク設定を確認。サーバのネットワークに問題がなければ、NIS サーバ上で以下を実行。(NIS サーバの起動)

```
$ cd /etc/rc.d/init.d
$ ./ypserv stop
$ ./ypserv start
```

この後再び、ステップ 2 を実行する。

7.2　ネットワークに接続できない？

1. `/usr/sbin/ifconfig -a` を実行する。
   (a) eth0 のエントリが無ければ、ネットワークデバイスが認識されていない。対処 2 でどうしようもなければ誰か詳しい人に聞く。
   (b) eth0 のエントリがあれば、設定に問題がある可能性大 (以下の対処方法リストの 2 参照)。

2. 研究室内のマシン（自分以外）に ping をかける。
   (a) ping が帰って来なかったら下記の**対処方法リスト** 1,2 を順に確認
   (b) ping が帰って来たら次に進む
3. 研究室外の同じセグメントのマシン (192.168.0.xxx, xxx は IP アドレスリストを見ていろいろためす) に ping をかける。
   (a) ping が帰って来ないなら**対処方法リスト** 2 で、特にルートアドレスの設定を確認する。
   (b) ping が帰ってくるなら次へ。
4. 大学外のマシンへのドメイン名での接続ができない場合。

   ネームサーバ (設定ファイル/etc/resolv.conf に書いてある nameserver の IP アドレス) に ping してみる。ping が帰って来ないならおそらくネームサーバが落ちてるので、他のネームサーバにアクセスするように `resolv.conf` を書き換えるか、誰かに助けを求める。

対処方法リスト

1. ネットワークケーブルがしっかり刺さっているか確認
2. network-admin コマンドで、IP アドレス等ネットワーク情報の確認後、インターフェース eth0 を起動する。

7.3 固まった!!!!!

なんらかの作業をしていて画面が固まってしまい、キー入力を受け付けなくなったときには、以下を順に試す。

1. 仮想画面に移って問題と思われるプロセスを kill する
2. 他の端末から、rsh や ssh などで入って、問題と思われるプロセスを kill する
3. X Window を立ち上げているときには {CM}-BackSpace を押すと X Window が終了できる。
4. 以上でダメなら、周りに助けを求める
5. **(初心者はやってはいけない)** 助けがなければ、数分様子を見てから、{CM}-Delete でシステムの終了を試みる
6. **HD が壊れるかもしれないが**, リセットボタンを押す。

索 引

Symbols

.bash_profile . . . 11
.bashrc . . . 11
/etc/apt/sources.list . . . 21
/etc/exports . . . 25
/etc/fstab . . . 28
/etc/resolv.conf . . . 30
$ . . . 13
$# . . . 13
${##} . . . 14
${#} . . . 14
${%%} . . . 14
${%} . . . 14
$0 . . . 13
$1 . . . 13

A

alias . . . 11
apt . . . 21
apt-cache . . . 22
apt-get . . . 21
autofs . . . 26

B

bg . . . 3
bunzip2 . . . 6
bzip2 . . . 6
bzip2dir . . . 6

C

C-c . . . 4
case . . . 16
cat . . . 5
cd . . . 5
chmod . . . 13
cp . . . 4

D

deb . . . 21
diff . . . 5

E

echo . . . 12
edquota . . . 28
emacs . . . 17
exit . . . 14
exportfs . . . 25

F

fg . . . 3
for . . . 15

G

GNOME . . . 2
grep . . . 5
gunzip . . . 6
gzip . . . 6
gzipdir . . . 6

H

history . . . 4
HOME . . . 12

I

if . . . 14
ifconfig . . . 29

J

jed . . . 17
jobs . . . 3

K

KDE . . . 2
kill . . . 5
killall . . . 6

L

less . . . 5
ln . . . 10
locate . . . 7
ls . . . 4
lv . . . 5

M

man . . . 9
Meta キー . . . 1
mkdir . . . 5
mount . . . 26
mv . . . 4

N

network-admin . . . 30
NFS . . . 25
ng . . . 17
NIS . . . 29
nkf . . . 7

P

PATH . . . 12
PID . . . 5
ping . . . 8
printenv . . . 12
ps . . . 5
pwd . . . 5

Q

quota . . . 27
quotaon . . . 28

R

region . . . 19
rm . . . 4
rmdir . . . 5
rpm . . . 21

S

scp . . . 8
setquota . . . 28
setup . . . 25
shutdown . . . 10
sort . . . 5
source . . . 12
ssh . . . 8

T

tail . . . 5
tar . . . 6
TeX . . . 20
top . . . 5, 6
touch . . . 5

U

unalias . . . 11

W

wc . . . 5
webmin . . . 25
which . . . 7

X

X Window System . . . 2
xkill . . . 6

Y

yatex . . . 20
ypwhich . . . 29
yum . . . 21

あ

ウィンドウマネージャ . . . 2
エイリアス . . . 11

か

仮想画面 . . . 2
強制終了 . . . 4, 5
コピー . . . 2
コマンド検索 . . . 4
コマンド履歴 . . . 4
コメント文 . . . 13
コンソール画面 . . . 2

さ

シェル . . . 11
シェルスクリプト . . . 12
ジョブ一覧 . . . 5
ジョブ番号 . . . 4
シンボリックリンク . . . 10

た

ディレクトリ使用制限 . . . 27

は

パイプ . . . 9
パス . . . 12
バックグラウンド処理 . . . 3
パッケージ . . . 21
引数 . . . 13
ファイルサーバ . . . 25
プロセス ID . . . 5
ペースト . . . 2

ま

文字変数 13

ら

リージョン 19
リダイレクト 9

わ

ワイルドカード 9